Fortify Cheat Sheet

This page has been made public for vendors

Introduction

The purpose of this document is to provide some notes that may be of assistance to VA application developers in order to get started scanning VA application source code using Fortify. This document is not a comprehensive reference for the Fortify product. Fortify product documentation (installed in the "\Docs" directory of VA developer installations of Fortify) should be consulted for clarification on finer points of using Fortify. Additionally, Fortify end user training is available.

Where Do Defects Related To Security Come From?

There are very specific reasons why software developers must take deliberate, specific actions both when writing new code and when maintaining legacy code in order to mitigate security-related defects. Mitigating software defects cannot be delegated to other information technology groups, and in many cases compensating security controls may be ineffective or defeatable by attackers. These types of software defects come from two sources:

- Design-related decisions made during application development, and
- Implementation decisions made during the actual coding of applications

Typically, half the defects in software applications result from design-related decisions made during development. An example of a design-related vulnerability is not making calls to security controls in code in the correct locations. A common vulnerability of this type is making access control checks on the client side, as opposed to the server side of a web application. Typically, the other half of the defects in software applications result from implementation decisions made during the actual coding of applications. An example of a coding-related vulnerability is not using programming interfaces that prevent control characters from being sent to downstream interpreters. A common vulnerability of this type is querying a relational database using queries that were constructed by concatenating strings, as opposed to using parameterized interfaces.

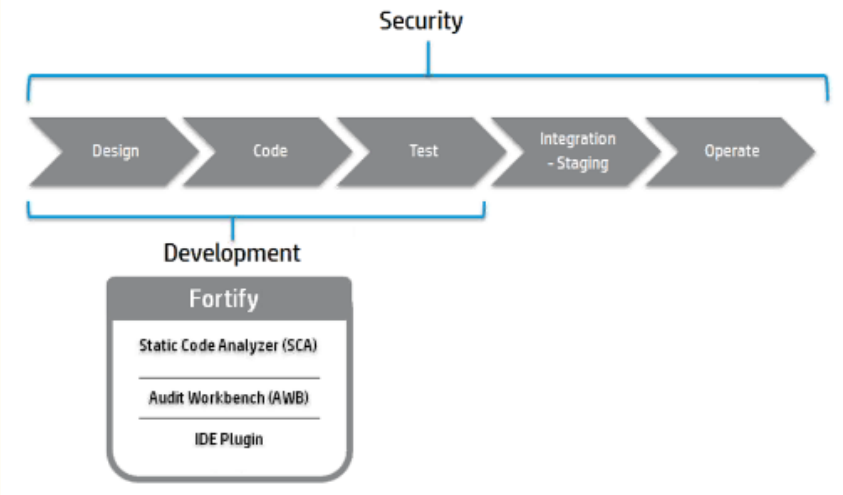
How Does Fortify Help Me Find, Fix, and Prevent Defects?

Fortify is a static analysis tool. This means that it can trace through your VA application source code and apply various types of rules as it does so in order to identify defects. This is as opposed to for example testing your VA application while it is running, or analyzing the architecture of your application. After a scan is completed, results are presented in a prioritized fashion and some guidance is provided to make fixes.

Overview of Fortify

There are various Fortify installation options that the VA is licensed for. Licensing options for Fortify in general mainly have to do with allowing the use of plug-ins that are available for some IDEs, and allowing the use of different scan rulepacks that are available for various programming languages. The Fortify product can be thought of as being made up of three components, as depicted in the figure below.

Table of Contents:



The Fortify Static Code Analyzer component is the engine that scans code. It can be called using the command line, using the Audit Workbench component, or using an IDE plug-in. The Fortify Audit Workbench component is a standalone application that provides a GUI to perform scans and to review scan results. The Fortify IDE plug-ins add capabilities to supported IDEs that are similar to the GUI functionality of the Audit Workbench.

An overview of the basic Fortify installation options that the VA is licensed for is provided below.

Standalone Installation

A standalone installation of Fortify can be used by VA developers separately than any tools that they may already have installed to develop applications. This installation includes for example a GUI front end called Audit Workbench that can be used to scan application source code and to review scan results. The VA has a license that supports standalone installations of Fortify on all platforms supported by Fortify, including Windows, Mac OS, and UNIX.

Optional Plug-Ins

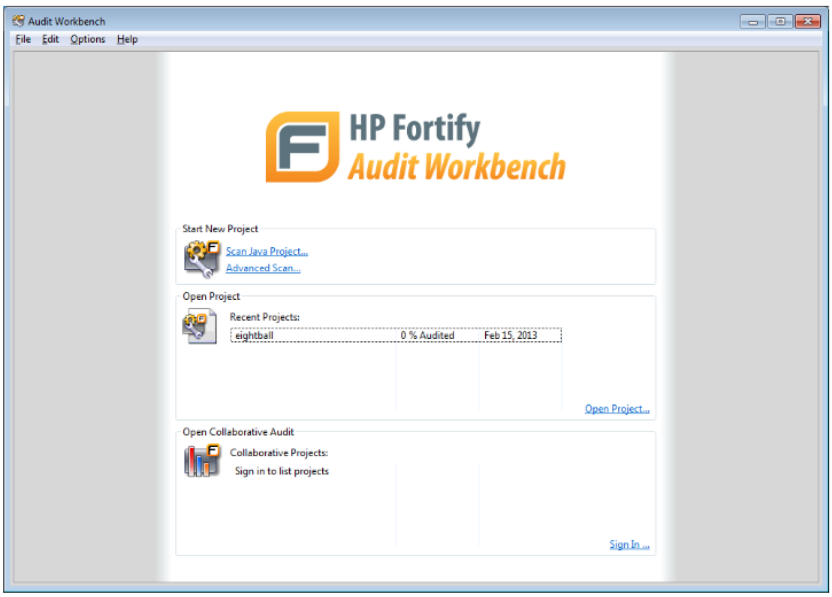
A standalone installation of Fortify can optionally include one or more plug-ins that can be used with certain tools that VA developers may already have installed to develop applications. The VA license includes Fortify plug-ins for Visual Studio and Eclipse. See the Fortify documentation for details.

Scanning a Hello World Project Using the Audit Workbench

Sample code that can be scanned using Fortify is installed by default as part of the standalone Fortify installation. The sample code that will be used as a "hello world" (i.e. first, simple) scan can be found in the "\Samples\basic\eightball" directory of VA developer installations of Fortify. It consists of a single Java language source code file.

Getting Started With The Audit Workbench

The Fortify Audit Workbench can be started from the Windows shortcut on VA developer desktops, or from the Windows Start menu. A screen snapshot of the default start-up screen of Fortify Audit Workbench is below.

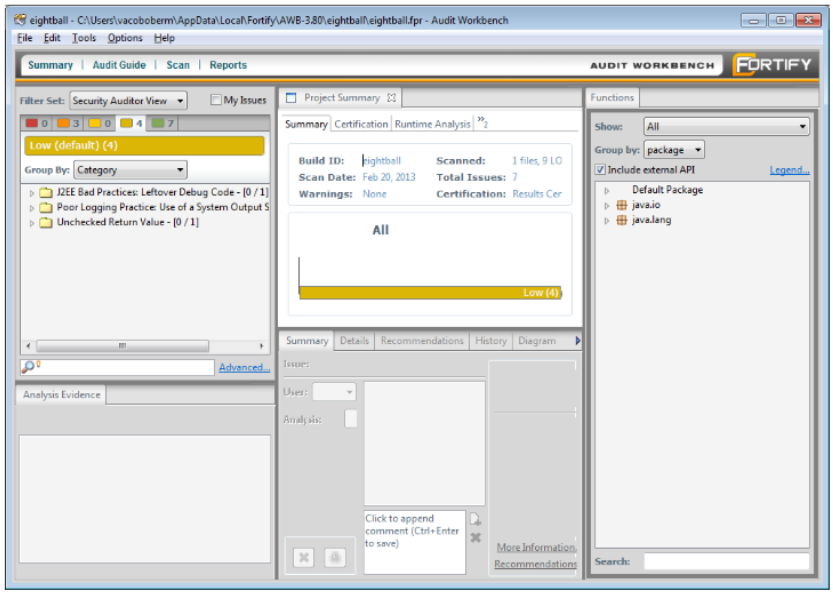


Scanning Hello World Source Code

The sample code can be scanned as follows:

- Select “Scan Java Project” •Select “C:\Program Files\HP_Fortify\Fortify380\Samples\basic\eightball” as project root
- Do not change default Java version
- Do not change default scan options
- Click “Run Scan” on “Audit Guide Wizard...”

After the scan completes, the Audit Workbench should look like the following screen snapshot:



Reviewing Scan Results

The scan results can be reviewed using various features of the Fortify Audit Workbench GUI. The features of the GUI that VA developers should familiarize themselves with include:

- The area in the upper left which can be used to filter and click through scan

results in a prioritized fashion,
- The area in the middle bottom which can be used to get additional details about an individual scan result, as well as suggestions for making fixes, and
- The area in the middle top which will scroll to the line of code that corresponds to a scan result as one clicks through the area in the upper left.

An example of selecting an individual scan result is below. Note how the file where the defect was identified is displayed, the line of code where the defect was identified within the file is highlighted, and how the Summary, Details, and other tabs in the area in the middle bottom is now populated.

Note that there are additional Fortify Audit Workbench capabilities that VA developers should additionally familiarize themselves with as time and priorities allow, particularly in the area of generating reports. There is an Audit Workbench User Guide that can be found in the "\Docs" directory of VA developer installations of Fortify as time and priorities allow.

Hints and Tips for Reviewing Scan Results

There are limits to the capabilities of static analysis tools in general to understand and analyze source code for various reasons. There will be results returned in some instances by Fortify that are in other words noise, i.e. that are not defects. Fortify scan results should always be reviewed for accuracy and completeness before for example generating metrics. For example, reporting a large number of what amount to informational findings can skew defect counts and unnecessarily cause concern.

Reviewing for False Positives

Reviewing Fortify scan results for issues that it reports that are not accurate is called reviewing for false positives. VA developers should always review scan results for false positives.

Reviewing for False Negatives

Additionally reviewing VA application source code manually or for example performing an architectural analysis is called reviewing for false negatives, i.e. defects that Fortify did not report. Reviewing for false negatives generally requires specialized expertise, for example by an experienced secure code reviewer. VA developers should review scan results for false negatives to the extent practical, e.g. if a developer is familiar with the correct use of a security library, to manually review code to ensure its correct use regardless of what Fortify reports.

Where To Go From Here

Additional VA resources that may be helpful to VA application developers can be found elsewhere on this site.